Deep Memory Network for Cross-Modal Retrieval

Ge Song, Dong Wang, and Xiaoyang Tan

***Abstract*—With the explosive growth of multimedia data on the Internet, cross-modal retrieval has attracted a great deal of attention in both computer vision and multimedia communities. However, this task is challenging due to the heterogeneity gap between different modalities. Current approaches typically involve a common representation learning process that maps data from different modalities into a common space by linear or nonlinear embedding. Yet, most of them only handle the dual-modal situation and generalize poorly to complex cases that involve multiple modalities. In addition, they often require expensive fine-grained alignment of training data among diverse modalities. In this paper, we address these with a novel cross-modal memory network (CMMN), in which memory contents across modalities are simultaneously learned from end to end without the need of exact alignment. We further account for the diversity across multiple modalities using the strategy of adversarial learning. Extensive experimental results on several large-scale datasets demonstrate that the proposed CMMN approach achieves state-of-the-art performance in the task of cross-modal retrieval.**

***Index Terms*—Cross-modal retrieval, memory network, deep learning.**

I. Introduction

RETRIEVING data from multiple modalities corresponding to one phenomenon is a useful technique to achieve the comprehensive knowledge of the phenomenon of interest, e.g., a sketch portrait of a criminal provided by the witness can be used to find relevant face images about that criminal in the criminal investigation. Information querying scenario like this is often called cross-modal retrieval (CMR) in literature [1]. This challenging task has gained increasing attention from both industrial and academic communities due to its wide usage in real-world applications.

Among others, two issues are central to the task of CMR: one is the measurement of content similarity among data from different modalities (also referred as the heterogeneity gap); the other is related to the storage and retrieval efficiency problem when dealing with large-scale multimedia data. Many cross-modal representation learning methods [2]–[6] are proposed and typically work by performing some mapping from different modalities to a unified feature space with various linear or non-linear transformations, such that different modalities of the data become computationally comparable. Despite the effectiveness of these methods, they have a few limitations: first, numerous of them only focus on two types of modalities (e.g., image-text) (using either traditional statistic analysis such Canonical Correlation Analysis (CCA) based methods [7], [8] or more recent deep models [3], [9]). Although several methods [5], [10] have extended this to handle more than two modalities, they are not very adaptive when a new modal is available; Second, many existing methods [2], [11], [12] require that the training data are exactly aligned among various modality at a fine-grained level, e.g., in the form of image-text pairs. This is not practical in the real-life scenario. For example, for two collections of news of video and audio sharing the same topics, it is hard if not impossible to match them at a very detailed level. Last but not least, prior knowledge about the underlying phenomenon of interest is not fully exploited in these methods, e.g., information about the type of modality and the corresponding expressive power of each type are generally ignored.

Due to the low storage costs and the high computational efficiency of binary codes, hashing based approximate nearest neighbors (ANN) search methods [13], [14] have achieved great success in image retrieval tasks and attracted increasing attention. Recently, this idea of hashing schemes [10], [15], [16] has been extended to cross-modality retrieval by embedding the data of interest into a low-dimensional Hamming space so as to efficiently preserve the cross-modality similarity. However such methods may suffer from the same limitations mentioned before in accommodating new modality and in aligning diverse modalities.

In this paper, we address the above issues with a newly-introduced deep network architecture with memory mechanism for cross-modal representation learning. The method is named Cross-Modal Memory Network (CMMN), and is motivated by the following thinking experiment: if we ask someone to imagine a scene of soccer game, he or she may first interpret in his/her brain what the soccer game is (e.g., 'Some people are playing football on the grass field') and search some impressive components in his/her memory (e.g., person, football), and then these components are aggregated, processed and transformed into the final image of the scene. Such procedure of integrating and mapping various attention contents relevant to the target object across various modalities are helpful for common

Manuscript received March 28, 2018; revised August 2, 2018 and September 3, 2018; accepted September 22, 2018. Date of publication October 24, 2018; date of current version April 23, 2019. This work was supported in part by the National Natural Science Foundation of China under Grants 61672280, 61373060, and 61732006, in part by the Pre-research fund of Equipments of China, in part by the Jiangsu 333 Project BRA2017377, in part by the Qing Lan Project, and in part by the Jiangsu Innovation Program for Graduate Education under Grant KYLX15_0320. The associate editor coordinating the review of this manuscript and approving it for publication was Dr. Xavier Giro-i-Nieto. *(Corresponding author: Xiaoyang Tan.)*

The authors are with the College of Computer Science and Technology, Nanjing University of Aeronautics and Astronautics, Nanjing 211106 China, and also with the Collaborative Innovation Center of Novel Software Technology and Industrialization, Nanjing 211106, China (e-mail: sunge@nuaa.edu.cn; dongwang@nuaa.edu.cn; x.tan@nuaa.edu.cn).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TMM.2018.2877122

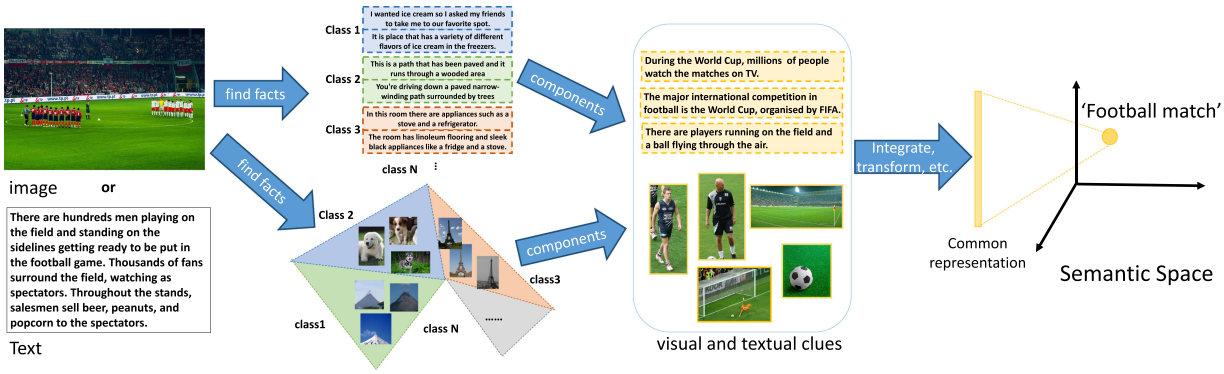


Fig. 1. Illustration of our motivation. We observed that heterogeneous data could be represented by a collection of corresponding related clues (visual objects in images, textual entities in texts or others) across different modalities. Initially, the features with the salient semantic concepts of each modal are pre-learned and stored in memory components of CMMN. For a given query, the CMMN automatically learns to search from memory supporting clues in different modalities, and the corresponding common representation of input is then obtained by aggregating and transforming these clues found.

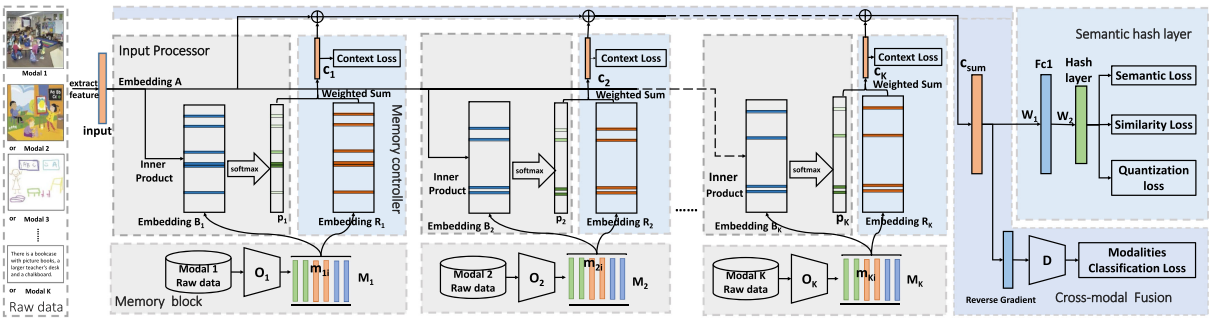


Fig. 2. Overview of the proposed cross-modal memory network. The proposed architecture consists of five components: Memory block (M), Input processor (I), Memory controller (C), Cross-modal fusion (F) and Semantic hash layer (H). It firstly pre-trains N individual classifiers O of semantic concept for each modal to initialize working memory contents in M. When the input goes through the network, N relevant context features c of each modal will be obtained with attention-based addressing of I and reading of C, and consequently are merged into an internal representation via F, where an extra classifier D of the modal type with adversarial learning strategy is incorporated to achieve fine-fusion. After that, this representation is converted into binary-like codes by semantic hash layer H for fast retrieval. Semantic and similarity loss is used to maximize the discriminative of learned code and the relevance of cross-modal data, and simultaneously quantization loss is imposed on the real-valued output to approximate the desired discrete values.

representation learning. Besides, the imaging way of searching useful components is quite similar with the data processing of memory neural network (MemNN). MemNN uses memory to store some facts in memory and retrieving supporting facts for input to do inferring. Therefore, we can take advantage of the memory mechanism to design a memory network that can learn to find supporting pre-stored clews (i.e., visual objects in images, textual entities in texts or others) of different modalities for the input data. These clews are related to the input data in semantic level, and then the input data and its relevant clews can be mapped into a common representation space through nonlinear transforming of CMMN.

The above idea is illustrated in Fig. 1, and a possible implementation using the proposed CMMN network is given in Fig. 2. We also use the category information of multi-modality to enhance the quality of fused common representation under the framework of adversarial learning and adopt additional hashing layer to accelerate the retrieval efficiency. Extensive experimental results show that the proposed CMMN approach achieves state-of-the-art performance on the Wikipedia [7], MIRFLICKR [17], NUS-WIDE [18] and Microsoft COCO [19] datasets and outperforms several other baselines on the large-scale scene dataset CMPlaces [20]. A preliminary version of this work appeared in [21].

The remainder of this paper describes and analyzes our CMMN in detail. The related work is described in Section II, the details of the CMMN model are presented in Section III, and experimental results are given in Section IV. We draw our conclusions in Section V.

II. Related Work

In this section, we briefly review the related methods for cross-modality retrieval and discuss some of the recent works concerning memory neural networks.

A. Cross-Modal Retrieval

During the past few years, many approaches [6]–[8], [22], [23] have been proposed for cross-modal retrieval. The key idea of these is to map heterogeneous data into a shared common representation space to account for the diversity of different modalities. One representative approach is the canonical corre-

lation analysis (CCA) [7] and its variation in modern days [5], [8], which learn two separate linear mappings for two corresponding modalities so that they are maximally correlated in one common subspace. Since these methods cannot easily capture high-level semantics simply from the raw visual or textual features, Gong *et al*. [5] present a three-view embedding method named TV-CCA, which incorporates a third semantic view into CCA. However, most of these methods focus on enhancing cross-modal similarity of aligned data and ignore the preservation of the intra-modal similarity.

Many methods [3], [4], [20], [23] investigate the possibility of learning complex nonlinear embedding networks based on deep architectures. For example, Srivastava *et al*. [24] used Deep Boltzmann Machine (DBM) to learn a probability density over the space of multimodal inputs and obtained a unified representation for each single-modal data, whereas Ngiam *et al*. [25] adopted several unsupervised deep models to learn features over multiple modalities. Feng *et al*. [26] proposed a correspondence autoencoder (cor-AE) network to maximize the correlation of hidden representations of two uni-modal autoencoders.

Recently more complex deep models [3], [4], [11], [20], [23], [27], [28] with supervised information have been employed for cross-modal learning. In [23], Jiang *et al*. presented a method that takes advantage of fine-grained local information of image patch and textual words to optimize a pairwise ranking function for cross-modal retrieval. To learn the cross-modal representation of weakly alignment scene data, Castrejon *et al*. [20] established a large-scale multi-modal scene dataset named CM-Places and presented a method to regularize the deep features from different modalities such that they share the same Gaussian distribution. Salvador *et al*. [3] also proposed a joint neural embedding model and used semantic regularization to learn a common representation. Wei *et al*. [4] further investigated the feasibility of using the off-the-shelf and fine-tuned CNN features to tackle cross-modal retrieval with semantic matching. Despite their effectiveness, all of the aforementioned methods assume the availability of a large number of matched aligned cross-modal pairs which are unfortunately not always available in practice.

Hashing techniques [29], [30] have been successfully applied in the retrieval tasks to encode high-dimensional features into compact binary codes, hence enabling extremely fast similarity search with Hamming distances. More recently many researchers have investigated the techniques of cross-modal hashing [16], [31]–[36], and various regression methods such as kernel logistic regression [10] and AdaBoost [37] have been used for this purpose. Zhang and Li [38] proposed to integrate semantic labels into the hashing learning procedure and learned the binary codes bit by bit. Jiang and Li [16] presented a method that integrates feature learning and hash learning in the same end-to-end deep learning framework and optimizes the model via maximizing the likelihood of the cross-modal similarities. Cao *et al*. [9] developed a method that generates compact hash codes of images and sentences using stacked LSTMs and CNN, whereas the deep sketch hashing method by Liu *et al*. [39] consists of three convolutional neural networks to encode free-hand sketches, natural images, and the auxiliary sketch-tokens.

B. Memory-Augmented Neural Network

The central idea of memory-augmented networks [40] is to enhance the network's long-term memory capability by augmenting it with a series of extra memory components. These memory components can be read and written to store input facts (statements sentence) and to retrieve supporting facts for an input query. As a specific implementation of this idea, Sukhbaatar *et al*. [41] developed an end-to-end neural network MemN2N with a recurrent attention model over a large external memory. Let $x = [x_1, \ldots, x_n]$ represent n facts with the associated embeddings $[\mathbf{x_1}, \ldots, \mathbf{x_n}]$ and let $\mathbf{q}$ be the embedding of the query q. To read from the memory, the posterior of each memory slot i being selected given the query $\mathbf{q}$ and the fact $[\mathbf{x_i}]$ in that slot is first estimated, and using this the corresponding supporting facts c stored in memory for q is then calculated in a soft attention manner.

Miller *et al*. [42] presented a key-value memory network which allows utilizing different encoding schemes for memory reading, and a similar memory mechanism is adopted in Neural Turing Machine (NTM) of [43] to tackle the problem of sorting and recalling. Note that the attention mechanism is popularly employed along with the reading and writing operation of the memory, which basically computes a categorical distribution so as to get a soft-selection over memory slots. Recently Kim *et al*. [44] have proposed a new structured attention network which instead used a conditional random field to capture such structural dependencies of memories.

The memory-augmented neural network is also widely used in the field of computer vision. For example, the stacked attention networks (SANs) [45] use the semantic representation of a query to search for the regions in a given image to infer the answer. In this scheme, the whole image is stored as memory, and it exploits supporting visual cues for reasoning. As another example, Iterative Querying Model (IQM) [46] encodes human-curated knowledge evidence into an extra memory bank as the auxiliary for more accurate reasoning. Meanwhile, some recent efforts [47], [48] have explored ways to use RNNs or LSTM-based models with memory in the natural language processing (NLP) field. Wang *et al*. [47] proposed to enhance the RNN decoder in neural machine translator with a pre-determined size external memory so that it can capture relevant information for the decoding process. Cheng *et al*. [48] extend the LSTM-based machine reader with a memory network, explicitly storing contextual information of input tokens to induce their relations and perform shallow reasoning.

It is worth mentioning that the above memory-augmented neural networks are used either for visual Question-Answer (QA) [45], [46] or for text understanding [47], [48] and only one type of modal is involved in these models, whereas our work extends these for cross-modal representation learning.

III. The Proposed Method

In this section, we give a detailed description of the proposed cross-modal memory network model (CMMN) and its possible extensions.

A. Cross-Modal Memory Networks

Our cross-modal memory network (CMMN) consists of five major components: an input processor, a memory block and its controller, a cross-modal fusion component, and a semantic hashing layer. Fig. 2 illustrates the overall architecture of CMMN. Particularly, the network first takes the input from each modality through the *input processor*, and with the help the *memory block* and the companying *memory controller*, the input processor maps the multi-modal data into points in a common feature space. To further account for the diversity, the *cross-modal fusion component* aggregates these into a single vector using an adversarial learning strategy. Finally, the fused vector is sent to the *semantic hashing layer* to be condensed into compact binary codes for efficient retrieval.

Suppose that we have K modalities with each dimension D_k. For each modality k, we have a memory block $\mathbf{M_k}$, which consists of N_{M_k} memory units, each of which is denoted as a vector m_i^k in $R^{1\times D_k}$.

Memory Block: Prototype concepts representation is critical for cross-modal learning in the proposed CMMN. Human beings always remember the general and specific characteristics of classes which are helpful to distinguish inter-class and intra-class objects. This intuition gives us some hints about what should memories store - those prototypes containing more general characteristic of the specific concept. We call our computational facilities that store prototypes 'memory block', one memory unit for one prototype. To learn them, for each modality, we use its N training data $\{(x_i, l_i\}_{i=1}^N\}$ (where $x_i \in R^{1\times D_k}$, $l_i \in \{0,1\}^{1\times C}$, and C is the total number of classes in that modality[1]) to train a classifier O that predicts the conditional probability $p(l_i|x_i)$. Then the content of memory for that modality can be computed as follows:

$$\mathbf{M_k} = \bigcup_{j=1}^{C} \{x \in R^{1\times D_k} \mid x \in \underset{x_i \in C_j}{TOP_T}(p(l_{i,j}=1|x_i))\} \quad (1)$$

where $l_{i,j} \in \{0,1\}$ is the j-th element of label vector l_i, and C_j is set of samples in category j. In words, candidate prototypes in each category are sorted according to its posterior probability in that category, and the top T candidates are selected as memory content (T is a user-defined parameter). Memories for other modalities are generated in the way.

Input processor: To account for the diversity of multiple modalities, given an input $q \in R^{1\times D}$ of some modality, we need to re-express it using some 'language' that is independent of any modality, and this is where our 'memories' of each modality come in. Particularly, for both the input q and the memory block of each modality, we first convert them into a V-dimensional common continuous space with embedding matrices $\mathbf{A} \in \mathbb{R}^{D\times V}$, $\mathbf{B_k} \in \mathbb{R}^{D_k\times V}$, respectively. Then the match score between q and i-th memory unit in the k-th memory block $m_{k,i}$ is calculated in the embedding space using the inner product operation, and the softmax attention mechanism is used to determine memories' position. Let variables z_k denote the position of $\mathbf{M_k}$ to be read according to the query. Then the distribution of z_k over the positions of memory units in M_k can be estimated as follows:

$$p(z_k = i \mid m_{k,i}, q) = softmax\left((m_{k,i}\mathbf{B_k})(q\mathbf{A})^{\mathrm{T}}\right) \quad (2)$$

where $softmax(x_i) = \frac{1}{Z}\exp(x_i)$, and $Z = \sum_j \exp(x_j)$.

Memory controller: As each memory unit of a memory block contains both certain degrees of semantic information and certain degrees of noise, to read them out for a given input q, they are first undergone a linear transformation defined by a matrix $\mathbf{R_k} \in \mathbb{R}^{D_k\times V}$. This resulted in a filtered vector (also called a 'context vector') c_k, as follows,

$$c_k = \sum_{i=1}^{N_{M_k}} p(z_k = i \mid m_{k,i}, q))(m_{k,i}\mathbf{R_k}) \quad (3)$$

To learn the filtering matrix $\mathbf{R_k}$, we use a context loss $\mathrm{J_{context}}$. Depending on the types of labels l of training samples available, the exact formation of $\mathrm{J_{context}}$ adopted is different. Particularly, if the label l is a multi-class output vector, we use the usual cross-entropy loss, while in the case of multi-label, the squared reconstructive loss is used instead. Let the output of the context loss layer be $I_c(c_k) \in \mathbb{R}^{1\times D}$, and the input vector be q. This squared version of $\mathrm{J_{context}}$ is defined as below:

$$\mathrm{J_{context}}(c_k) = \|I_c(c_k) - q\|_F^2 \quad (4)$$

where $\|\cdot\|_F$ is the Frobenius norm of vector.

Cross-modal fusion: This layer links the K filtered vector $c_k \in \mathbb{R}^{1\times V}$ for the input q cross all the K modalities. To fuse these vectors, all filtered vectors c_k and the embedded vector of the input q are integrated into a vector c_{sum} by weighted summing, which is then mapped into a common representation space via a nonlinear transformation layer Fc1 (see Fig. 2). The final fused feature r is the output of Fc1:

$$\begin{aligned} c_{sum} &= (\alpha(q\mathbf{A}) + \sum_{j=1}^{K} \beta_j \mathrm{c_j}) \\ r &= relu(c_{sum}\mathbf{W_1} + b_1) \end{aligned} \quad (5)$$

where $\mathrm{relu}(x) = \max(x, 0)$, $\mathbf{W_1} \in R^{V\times D_1}$ and b_1 are the weight and bias of Fc1 layer, D_1 is the number of nodes of Fc1. The parameters $\alpha, \{\beta_j\}_{j=1}^K$ are initialized to be 1 and are learnt during the training procedure with back-propagation.

Intuitively, a good modality fusion model should have two characteristics: one is to preserve as much semantic information as possible, and the other is that after fusing it should weaken the trace of the original modality as much as possible (as this indicates that the information has been well fused). We achieve these two goals within the adversarial learning framework [49].

Assuming that each sample is associated with a modality type label $l_{mod} \in \{0,1\}^{1\times K}$ (if the sample comes from the j-th modality then l_{mod_j} is 1, otherwise is 0). We can introduce a modality classifier (D) to measure the fusion quality of c_{sum},

[1]Here for simplicity we only consider the situation where both the category types and the number of categories of data in each modality are kept the same.

the goal of D is to minimize the following function:

$$\mathrm{J}_{\mathrm{classifier}}(c_{sum}, l_{mod}) = -\sum_{j=1}^{K} l_{mod_j} \log(\hat{l}_{mod_j}) \tag{6}$$

where $\hat{l}_{mod}$ is the predicted modality label of c_{sum} through the classifier D.

Based on above intuition, the goal of CMMN is to adjust its weights to weaken the trace of the original modality in c_{sum} as much as possible, that is to maximize Eq.(6). This procedure can guide CMMN to search more invariant and complementary memory contents for cross-modal fusion. Now we have two modules playing an adversarial game:

$$\max_{CMMN_D} \min_{D} \mathrm{J}_{\mathrm{classifier}}(c_{sum}, l_{mod}) \tag{7}$$

where $CMMN_D$ denots the part of CMMN before D.

To solve Eq. (7), we insert a gradient reserved layer before D (as shown in Fig. 2) to reverse gradient back-propagated from $\mathrm{J}_{\mathrm{classifier}}$ loss, The optimization process of CMMN and D with gradient descent can be roughly denoted as follows:

$$\begin{aligned}
\theta^{\mathrm{t}+1}_{\mathrm{CMMN_D}} &= \theta^{\mathrm{t}}_{\mathrm{CMMN_D}} - \rho \frac{\partial c_{sum}}{\partial \theta^{\mathrm{t}}_{\mathrm{CMMN_D}}} \left(-\frac{\partial \mathrm{J}_{\mathrm{classifier}}}{\partial c_{sum}} \right) \\
&= \theta^{\mathrm{t}}_{\mathrm{CMMN_D}} + \rho \frac{\partial \mathrm{J}_{\mathrm{classifier}}}{\partial \theta^{\mathrm{t}}_{\mathrm{CMMN_D}}} \\
\theta^{\mathrm{t}+1}_{\mathrm{D}} &= \theta^{\mathrm{t}}_{\mathrm{D}} - \rho \frac{\partial \mathrm{J}_{\mathrm{classifier}}}{\partial \theta^{\mathrm{t}}_{\mathrm{D}}}
\end{aligned} \tag{8}$$

where $\theta^{\mathrm{t}}_{\mathrm{CMMN_D}}$ and $\theta^{\mathrm{t}}_{\mathrm{D}}$ are parameters of $CMMN_D$ and D at time t, respectively. And ρ is learning rate.

Semantic hashing layer: To speed up retrieval, the fused representation $r \in R^{1\times D_1}$ is transformed into D_2-dim binary codes by a hash-layer (a fully-connect layer activated with sigmoid or tanh function) with hashing constraints [13]. Let h be the output of hash layer.

$$h = \tanh(r\mathbf{W_2} + b_2) \tag{9}$$

where $\tanh(x) = \frac{(e^x - e^{-x})}{(e^x + e^{-x})}$, $\mathbf{W_2} \in R^{D_1 \times D_2}$ and b_2 are the weight and bias of the hash layer. Hash codes can be obtained by simple quantization $b = sign(h) \in \{-1, 1\}^{1\times D_2}$.

B. Loss Function and Optimization

To learn the proposed CMMN network, several heuristics are incorporated into the loss function. Besides the context loss $\mathrm{J}_{\mathrm{context}}$ and the modality classification loss $\mathrm{J}_{\mathrm{classifier}}$ described in the previous section, we adopted two additional loss terms that are helpful to improve the discriminative power of the learnt representation and to better capture the semantic similarity between two points (see Fig. 2):

Semantic preservation loss $\mathrm{J}_{\mathrm{semantic}}$*:* Given the binary codes b of the input q and its label l, if q is multi-class data ($\|l\|_1 = 1$), the $\mathrm{J}_{\mathrm{semantic}}$ is defined as follows:

$$\mathrm{J}_{\mathrm{semantic}}(b, l) = -\sum_{j=1}^{C} \left(l_j \log(\hat{l_j}) \right) \tag{10}$$

where $\hat{l}$ is the predicted label of CMMN. If q is multi-label data ($\|l\|_1 > 1$), the $\mathrm{J}_{\mathrm{semantic}}$ is defined as:

$$\mathrm{J}_{\mathrm{semantic}}(b, l) = -\sum_{j=1}^{C} \left(w \cdot l_j \log(\hat{l_j}) + (1 - l_j) \log(1 - \hat{l_j}) \right) \tag{11}$$

where w is the ratio of negative and positive samples.

Pairwise similarity loss $\mathrm{J}_{\mathrm{similarity}}$*:* For a pair of binary codes b_i and b_j, as there exists a linear relationship between their Hamming distance $d_H(\cdot,\cdot)$ and inner product $\langle \cdot, \cdot \rangle$: $d_H(b_i, b_j) = \frac{1}{2}(D_2 - \langle b_i, b_j \rangle)$, one can use the inner product as a surrogate of the Hamming distance [50]. Given a similarity measure s_{ij} (if $l_i \cap l_j \neq \emptyset$ then $s_{ij} = 1$, otherwise $s_{ij} = 0$) in the label space between the two points, the likelihood $p(s_{ij}|b_i, b_j)$ is defined as follows:

$$\begin{aligned}
p(s_{ij}|b_i, b_j) &= \begin{cases} \sigma(\langle b_i, b_j \rangle) & s_{ij} = 1 \\ 1 - \sigma(\langle b_i, b_j \rangle) & s_{ij} = 0 \end{cases} \\
&= \sigma(\langle b_i, b_j \rangle)^{s_{ij}} (1 - \sigma(\langle b_i, b_j \rangle))^{1 - s_{ij}}
\end{aligned} \tag{12}$$

where $\sigma(x) = \frac{1}{1+\exp(-x)}$ is the sigmoid function. Then the pairwise cross-entropy loss $\mathrm{J}_{\mathrm{similarity}}$ is defined as the negative log likelihood $p(s_{ij}|b_i, b_j)$:

$$\begin{aligned}
\mathrm{J}_{\mathrm{similarity}}(s_{ij}, b_i, b_j) = &\log\left(1 + \exp\left(\frac{1}{2} b_i^T b_j\right)\right) \\
&- s_{ij}\left(\frac{1}{2} b_i^T b_j\right)
\end{aligned} \tag{13}$$

Finally, the overall loss function for N training samples is defined as follows:

$$\begin{aligned}
\mathcal{J} = &\sum_{i=1}^{N} \mathrm{J}_{\mathrm{semantic}}(b_i, l_i) + \sum_{i=1}^{N}\sum_{j=1}^{N} \lambda_1 \mathrm{J}_{\mathrm{similarity}}(s_{ij}, b_i, b_j) \\
&+ \lambda_2 \sum_{i=1}^{N} \mathrm{J}_{\mathrm{classifier}}(c_{sum_i}, l_{mod_i}) + \lambda_3 \sum_{i=1}^{N}\sum_{j=1}^{K} \mathrm{J}_{\mathrm{context}}(c_{ij})
\end{aligned} \tag{14}$$

where c_{ij} denotes the j-th modal context vector of i-th sample. $\lambda_1, \lambda_2, \lambda_3$ are balance parameters. For multi-class problem, we use Eq. (10) for $\mathrm{J}_{\mathrm{semantic}}$, and the softmax loss for $\mathrm{J}_{\mathrm{context}}$; while for the multi-label problem, $\mathrm{J}_{\mathrm{semantic}}$ is given in Eq. (11), and $\mathrm{J}_{\mathrm{context}}$ takes the form of Eq. (4).

As the problem of Eq. (14) is a discrete optimization problem, which is NP-hard to solve, we have to make some relaxation by replacing the binary codes b with the $tanh$ activation h of the hash layer (see Eq. (9)). However, this will give rise to some approximation error, i.e., $||b - h||_1$. To control this quantization error, we introduce another loss term $\mathrm{J}_{\mathrm{quantization\_loss}} = ||b - h||_2^2$, which enforces h to be saturated. Hence the final objective

Eq. (15) is rewritten as follows:

$$\mathcal{J} = \sum_{i=1}^{N} \mathrm{J}_{\mathrm{semantic}}(h_i, l_i) + \sum_{i=1}^{N}\sum_{j=1}^{N} \lambda_1 \mathrm{J}_{\mathrm{similarity}}(s_{ij}, h_i, h_j) + \lambda_2 \sum_{i=1}^{N} \mathrm{J}_{\mathrm{classifier}}(c_{sum_i}, l_{mod_i}) + \lambda_3 \sum_{i=1}^{N}\sum_{j=1}^{K} \mathrm{J}_{\mathrm{context}}(c_{ij}) + \lambda_4 \sum_{i=1}^{N} \mathrm{J}_{\mathrm{quantization\_loss}}(h_i, b_i) \quad (15)$$

where λ_1, λ_2, λ_3, λ_4 are balance parameters.

Because the overall CMMN model from input to output is smooth when b is fixed, it is easy to compute gradients and back-propagate through it. In CMMN, the parameters mainly lie in three parts, the embedding matrix $\mathbf{A}$ for input feature, two embedding matrices $\mathbf{B}, \mathbf{R}$ for each memory block, and other mapping matrices $\mathbf{W_1}, \mathbf{W_2}$. All these parameters including the memory block $\mathbf{M_k}$ for each modality are jointly learned by minimizing the objective loss (15), which is performed using stochastic gradient descent (SGD). At each iteration, the current version of binary codes b is first updated by thresholding the output of the hashing layer of the CMM, and then it is kept fixed until the next round of updating.

C. Extensions

Accommodating new modality: The proposed CMMN framework is not only able to handle multiple modalities at the same time, but it can also conveniently accommodate new modality if needed, as it does not need fine-grained modality alignment. Particularly, the following steps need to be performed for a new modality. First, the corresponding memory $\mathbf{M_{K+1}}$ is generated using Eq. (1). Next, it will be assembled to CMMN by adding a set of model parameters (i.e., $\mathbf{B_{K+1}}, \mathbf{R_{K+1}}, \beta_{K+1}$) for it. In the training stage, a new context vector c_{k+1} will be calculated with Eq. (3) and then fused into the learned representation with Eq. (5), whose discriminability could be further enhanced using the adversarial training method described in the previous section, if needed.

Using aligned data: Although the whole procedure of our CMMN method needs no explicit modality alignment, sometimes data with aligned modalities can be relatively easy to obtain. For example, on many web pages, images often co-occur with text descriptions. In such cases, it is useful to exploit the available alignment information to further boost the discriminative property of learned codes. Here, a modified probabilistic approach [10] is adopted. Assume a sample is represented by K alignment modalities features $\{x^{(1)}, x^{(2)}, \ldots, x^{(K)}\}$, where $x^{(k)} \in R^{D_k}$. Their modality-specific predicted binary-like codes are denoted as $\{h^{(1)}, h^{(2)}, \ldots, h^{(K)}\}$ and the corresponding hashing codes as $\{b^{(1)}, b^{(2)}, \ldots, b^{(K)}\}$, where $h^{(k)} \in (-1, 1)^{D_2}$, $b^{(k)} \in \{-1, 1\}^{D_2}, k = 1, \ldots, K$. The desired fused hashing codes of the sample is then denoted as $b \in \{-1, -1\}^{D_2}$.

With the data points $\{x^{(1)}, \ldots, x^{(K)}\}$ that aligned across all the modality, we calculate the value of each bit of the hashing codes b by comparing the relative posterior probability of that bit being positive or not, as follows,

$$b_j = sign\Big(p(b_j = 1|x^{(1)}, \ldots, x^{(K)}) - p(b_j = -1|x^{(1)}, \ldots, x^{(K)})\Big) \quad (16)$$

By applying the Bayes' theorem and assuming that different modalities are conditionally independent on b_j, and that $p(b_j = 1) = p(b_j = -1)$, we have,

$$b_j = sign\left(\prod_{k=1}^{K} p(b_j^{(k)} = 1|x^{(k)}) - \prod_{k=1}^{K} p(b_j^{(k)} = -1|x^{(k)})\right) \quad (17)$$

where the value of that bit in each modality is estimated based on the corresponding bit of the output of the hashing layer,

$$p(b_j^{(k)} = 1|x^{(k)}) = \frac{h_j^{(k)} + 1}{2} \quad (18)$$

Under the situation of two aligned features $x^{(1)}$ and $x^{(2)}$, Eq. (17) can be simplified to be,

$$\begin{aligned} b_j &= sign(p(b_j^{(1)} = 1|x^{(1)})p(b_j^{(2)} = 1|x^{(2)}) \\ &\quad - p(b_j^{(1)} = -1|x^{(1)})p(b_j^{(2)} = -1|x^{(2)})) \\ &= sign(p(b_j^{(1)} = 1|x^{(1)}) + p(b_j^{(2)} = 1|x^{(2)}) - 1) \\ &= sign(h_j^{(1)} + h_j^{(2)}) \end{aligned} \quad (19)$$

IV. Experiments

A. Datasets and Experimental Settings

To evaluate the performance of the proposed CMMN method in cross-modal retrieval task, we conduct extensive experiments on five datasets, i.e., Wikipedia [7], MIRFLICKR [17], NUS-WIDE [18], Microsoft COCO [19], and CMPlaces [20]. Note that these datasets are collected not particularly for the task of cross-modal retrieval, but they do have the evaluation protocol for other vision tasks such as object recognition or multimodal data fusion. In our experiments, we randomly sample a portion of data from the training set to train models, while in the testing stage, for each modality, we use the whole training set as the dataset to be retrieved and use testing set as queries.

Wiki: is an image-text dataset generated from Wikipedia and consists of 2,173 training and 693 testing image-text pairs. For each pair, the image is represented by the 128-dimensional SIFT descriptor vector, and the 10-dimensional vector derived from a latent Dirichlet allocation (LDA) model gives the text description. Each pair associated with one of 10 semantic labels including history, biology and so on. We use the whole training set for training.

MIRFLICKR: is an image-text dataset and originally contains 25,000 pairs. Each pair associates with some of 24 labels. For pretreatment, we remove pairs without textual tags or annotated labels, and we subsequently get 18,006 pairs as the training set and 2,000 pairs as the testing set. We represent each image as a 2,048-dimensional deep feature extracted from the ResNet [51]

pre-trained on the ImageNet. The 1386-dimensional bag-of-words vector gives the text description. We sampled 5,000 pairs of the training set for training.

NUS-WIDE: contains 260,648 web images, and some images associate with textual tags, belonging to 81 concepts. Following [10], [16], only the top 10 most frequent labels and the corresponding 186,577 text-image pairs are kept. In our experiments, 80,000 pairs and 2,000 pairs are sampled as the training and testing sets respectively. We represent each image as a 2,048-dimensional deep feature extracted from the ResNet [51] network pre-trained on the ImageNet. The 1000-dimensional bag-of-words vector gives the text description. We sampled 5,000 pairs of the training set for training.

Microsoft COCO: is a large-scale object dataset, containing 82,783 training and 40,504 testing images. Each image is associated with five sentences (only the first sentence is used in our experiments), belonging to 80 most frequent categories. After pruning images with no category information, we obtained 82,081 image-sentence pairs as the training set. We represent each image as a 2,048-dimensional deep feature extracted from the ResNet [51] network pre-trained on the ImageNet. The 4800-dimensional Skip-thought vector [52] gives the sentence description. We sample 10,000 pairs of the training set for training and 4,956 pairs from the testing set as queries.

CMPlaces: is a large-scale places dataset that consists of five modalities. It includes 2.4 million training and 20,500 testing natural images (NAT), 14,830 training and 2,050 testing line drawings (LDR), 9,752 training and 2050 testing textual descriptions (DSC), 11,372 training and 1,954 testing clip art (CLP), 456,300 training and 2,050 testing synthetic Spatial text images (SPT). Each sample associated with a unique label of 205 scene categories. The average-pooling the 4800-D Skip-thought vectors [52] of each sentence give the DSC description. For pixel-based modalities (e.g., NAT, LDR, etc.), we separately fine-tuned the AlexNet (pre-trained on the Place 205 dataset [53]) on the corresponding training data and extract the 4,096-D feature from the `fc7` layer as the representation. We sample 38,950 examples of NAT and 18,450 examples of SPT from corresponding training sets, and we then mix them with all training examples of LDR, CLP, DSC modalities for training. We sample 1,000 examples from the testing set of each modality as queries.

B. Implements Details

We implement the proposed CMMN using Tensorflow.[2] Note that, to fill in CMMN, the dimension of input from different modalities is transformed to be equal, and the conditional probability $p(l|x)$ in Eq. (1) is estimated at the same time. This task is fulfilled with the classifier O_k of each modality (see Fig. 2). The structure of these preprocessing networks (i.e., O_k s) and the CMMN network are detailed as follows.

Data preprocessing: For the Wiki dataset, two MLP classifiers are separately trained on the corresponding training sets of image and text descriptions. D is the dimension of the input data. The structure is defined as follows: $input(D) \rightarrow fc1(D) \rightarrow fc2(42364) \rightarrow 13 \times 13 \times 256 \rightarrow maxpool2(size: 3 \times 3, stride: 2) \rightarrow fc3(4096) \rightarrow fc4(4096) \rightarrow fc5(C) \rightarrow softmax\ loss$, where $fc(*)$ is the fully-connected layer with $(*)$ nodes. All layers are activated by ReLU, and C is the number of classes. The outputs of *fc4* are used in CMMN as the feature of each modality, and the outputs of *fc5* are the estimation of $p(l|x)$.

For the Microsoft COCO dataset, the MLP structure is defined as follows: $input(D) \rightarrow fc1(2048) \rightarrow fc2(2048) \rightarrow fc3(2048) \rightarrow fc4(C) \rightarrow cross\ entropy\ loss$, all layers are activated by ReLU. The outputs of *fc3* are the preprocessed features. The outputs of *fc4* are $p(l|x)$.

For the MIRFLICKR and NUS-WIDE datasets, the MLP structure is defined as follows: $input(D) \rightarrow fc1(1024) \rightarrow fc2(1024) \rightarrow fc3(1024) \rightarrow fc4(C) \rightarrow cross\ entropy\ loss$, and all layers are activated by tanH . The outputs of *fc3* are the preprocessed features. The outputs of *fc4* are $p(l|x)$.

For the CMPlace dataset, following [20] did, for each type of pixel-level data, i.e., NAT, CLP, LDR, and SPT, we use the output of *fc7* of the fine-tuned Alexnets for features extraction, and the output of *fc8* of the fine-tuned Alexnets as estimation of $p(l|x)$. For DSC modality, we adopt the same structure of MLP as that of Wiki to preprocess Skip-thought vectors.

Configuration of CMMN: The detailed configurations of CMMN network are given in Table I. The '$\mathrm{fc_A}(256)$', '$\mathrm{fc_B}(256)$' and '$\mathrm{fc_R}(256)$' are fully-connected layers with 256 nodes. 'fc (n), $*$' denotes the fully-connected layer of n nodes with $*$ activation function.

In the training phase, the weights of layers are initialized by a Gaussian distribution. For Wiki, the model is trained with learning_rate $= 0.001$, batch_size $= 64$, dropout $= 0.8$ and epoches $=$ 500. For Microsoft COCO, learning_rate $= 0.001$, batch_size $=$ 64, dropout $= 0.6$ and epoches $= 300$. For MIRFLICKR, learning_rate $= 0.01$, batch_size $= 64$, dropout $= 0.8$ and epoches $=$ 300. For NUS-WIDE, learning_rate $= 0.01$, batch_size $= 64$,

TABLE I
CONFIGURATION OF THE PROPOSED CMMN

CMMN configuration for Wiki dataset				
Memory data 1		Input	Memory data 2	
$\mathrm{fc_{R_1}}(256)$	$\mathrm{fc_{B_1}}(256)$	$\mathrm{fc_A}(256)$	$\mathrm{fc_{B_2}}(256)$	$\mathrm{fc_{R_2}}(256)$
fc(256), relu	inner product	-	inner product	fc(256), relu
-	softmax	-	softmax	-
weighted sum		-	weighted sum	
fc (10)	weighted sum			fc (10)
gradient reverse		fc1 (1024), relu, dropout		
fc (2), softmax		fc2 (bit length), tanh		
-		fc (10), softmax		

CMMN configuration for Microsoft COCO dataset				
Memory data 1		Input	Memory data 2	
$\mathrm{fc_{R_1}}(256)$	$\mathrm{fc_{B_1}}(256)$	$\mathrm{fc_A}(256)$	$\mathrm{fc_{B_2}}(256)$	$\mathrm{fc_{R_2}}(256)$
fc(256), relu	inner product	-	inner product	fc(256), relu
-	softmax	-	softmax	-
weighted sum		-	weighted sum	
fc (2048)	weighted sum			fc (2048)
gradient reverse		fc1 (2048), relu, dropout		
fc (2), softmax		fc2 (bit length), tanh		
-		fc (80), sigmoid		

[2]http://www.tensorflow.org

dropout = 0.8 and epoches = 200. For CMPlace, learning_rate = 0.01, batch_size = 32, dropout = 0.5 and epoches = 300. Other setting is discussed in Section IV-C.

Evaluation Protocols: We perform cross-modal retrieval mainly with two kinds of tasks, which are defined as follows.

- *Image vs. Text (I vs. T):* Retrieve relevant data in the text training set using an image query.
- *Text vs. Image (T vs. I):* Retrieve relevant data in the image training set using a text query.

For multi-class datasets Wiki and CMPlaces, we consider two points are similar if they belong to the same category. We adopt the commonly-used Mean Average Precision (mAP) as the performance metric.

$$\mathrm{P@n} = \frac{\#\left\{\text{relevant images in top N results}\right\}}{\mathrm{N}}$$

$$\mathrm{AP} = \frac{\sum_{\mathrm{n}} \mathrm{P@n} \times \mathrm{I}\left\{\text{image n is relevant}\right\}}{\#\left\{\text{retrieved relevant image}\right\}}$$

$$\mathrm{mAP} = \frac{1}{\mathrm{Q}} \sum_{\mathrm{i}} \mathrm{AP_i} \tag{20}$$

where # is a count function, I is an indicator function, and Q represents the total number of queries. Notably, the mAP is computed from the retrieval list over the whole database set, while $\mathrm{mAP@n}$ calculates from the top n retrieval results. We also give the precision-recall curves.

For multi-label datasets Microsoft COCO, MIRFLICKR and NUS-WIDE, we follow [16] and compute MAP based on the criterion that it is regarded as a relevant result if the retrieved result shares at least one class label with the query. We also adopt Normalized Discounted Cumulative Gain (NDCG) [54] as the performance metric, which is defined as:

$$NDCG@p = \frac{1}{Z} \sum_{i=1}^{p} \frac{2^{r_i} - 1}{\log(1+i)} \tag{21}$$

where Z is the ideal $DCG@p$ and calculated form the correct ranking list. $r(i) = |l_q \cap l_i|$ denotes the similarity between the i-th point and the query. l_q and l_i denote the label set of the query and i-th position point.

Particularly, for Microsoft COCO dataset, following the retrieval evaluation metrics [55], we report the median rank (**Med** r) of the first retrieved closest ground-truth sentence or image. Here, we define the returned result that has the same label of the query as the closet ground-truth.

C. Parameter Setting and Analysis

In this section, we analyze the effect of memory size T and balance weights $\lambda_1, \lambda_2, \lambda_3, \lambda_4$ in Eq. (15). We initially set $\{T, \lambda_1, \lambda_2, \lambda_3, \lambda_4\}$ to {10, 0.1, 0.01, 0.1, 0.001} for Wiki, {30, 0.1, 0.01, 0.001, 0.001} for MIRFLICKR, {100, 0.1, 0.08, 0.001, 0.001} for NUS-WIDE, and {100, 1, 0.01, 0.1, 0.001} for MS COCO. Then, we separately tune them with other parameters fixing and report the performance in Fig. 3.

From Fig. 3(a) we observe that the retrieval performance increases firstly and then fluctuates within a certain range with the increase of T. This result indicates that an appropriate size of memories can help to learn cross-modal representation better, while a larger T means selecting less discriminative data as memory and may be useless.

From Fig. 3(b) we see that the CMMN method achieves the best performance at a certain value, since a smaller λ_1 may cause that the similarity structure cannot be captured effectively, while a larger λ_1 may enlarge the noise of similarity and reduce the discriminative of learned feature. We also observe similar results for the parameters λ_3 and λ_4. A suitable λ_4 is useful to reduce the quantization loss and make the learned feature near to the binary code, while a large λ_4 may lead the optimization process to focus less on preserving similarity.

Besides, the CMMN method obtains better performance when λ_2 is less than a specific value. However, if λ_2 is too large, it will hurt the discriminative of learned feature and reduce performance.

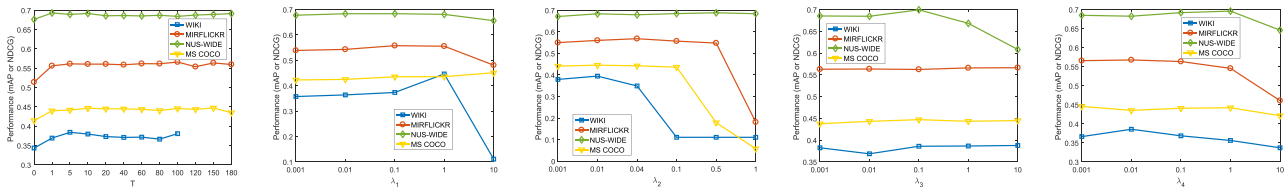
As we can see the proposed method is not sensitive to hype-parameters. We recommend user to set λ_1 in $[0.1, 1]$, λ_2 in $[0.01, 0.05]$, and λ_3, λ_4 in $[0.001, 0.1]$. In the following experiments, we empirically set $T, \lambda_1, \lambda_2, \lambda_3, \lambda_4$ to $\{100, 1, 0.1, 1, 0.001\}$ for MS COCO, to $\{10, 1, 0.001, 0.1, 0.001\}$ for Wiki, to $\{30, 0.1, 0.01, 0.01, 0.001\}$ for MIRFLICKR, to $\{100, 0.1, 0.01, 0.01, 0.001\}$ for NUS-WIDE. In particular, we set memory size T according to the data scale of different modalities for CMPlace. We set it to 10 for CLT, DSC, LDR and SPT modalities, and 90 for NAT. We set $\lambda_1, \lambda_2, \lambda_3, \lambda_4$ to $\{0.1, 0.001, 0.1, 0.001\}$.

D. Method Comparison

1) Compared Methods: To validate the superior of our CMMN method, we compare CMMN with other real-valued and binary-valued cross-modal representation learning methods [1]. We use Cosine distance to measure the similarity of real-valued feature and use Hamming distance to measure the similarity of binary-valued features.

For Wiki, Microsoft COCO, MIRFLICKR and NUS-WIDE datasets, we compare with real-valued methods TV-CCA [5], LCFS [56], JFSSL [2], Deep-SM [4], and hash methods CMSSH [31], CVH [32], IMH [33], CMFH [35], SCM [38], SePH [10], corAE [26], DSH [13], DCMH [16]. DSH is a unimodal deep hashing, which uses the contrastive loss to optimize a siamese network, we modify it for inputting two modalities. DCMH is similar to CMMN, which uses the pairwise loss to learn cross-modal similarity. We carefully implement the DSH and DCMH methods and replace their CNN sub-structures with the same MLP of the CMMN method for 2,048 ResNet features.

Because CMPlace dataset contains five unaligned modalities data and most previous approaches are designed for two modalities or require aligned data for training, we only compare CMMN with SePH [10] and deep baselines BL [20], $\mathrm{Deep_A}$ [20], $\mathrm{Deep_B}$ [20], $\mathrm{Deep_C}$ [20], Deep-SM [4]. Unlike CMMN, $\mathrm{Deep_A}$, $\mathrm{Deep_B}$ and $\mathrm{Deep_C}$ impose different regularization terms to make the output of modal-specific networks statistically similar. For a fair comparison with Deep-SM, we use the output of the top layer of the network as the semantic representation and perform semantic-matching in $\mathrm{CMMN_{SM}}$.



(a) (b) (c) (d) (e)

Fig. 3. Effect of parameters on the CMMN method. (a) illustrates the effects of Memory size T. (b) shows the effects of λ_1. (c) shows the effects of λ_2. (d) shows the effects of λ_3. (e) shows the effects of λ_4.

TABLE II
THE COMPARISON OF DIFFERENT REAL-VALUED REPRESENTATION LEARNING METHODS ON THE WIKI DATASET

Method	Wiki (MAP)			MS COCO (NDCG)			Mirflickr (NDCG)			NUS-WIDE (NDCG)		
	I *vs.* T	T *vs.* I	Avg	I *vs.* T	T *vs.* I	Avg	I *vs.* T	T *vs.* I	Avg	I *vs.* T	T *vs.* I	Avg
TV-CCA+CNN	0.2890	0.4966	0.3928	0.2693	0.2442	0.2567	0.3033	0.3034	0.3034	0.5129	0.5050	0.5090
LCFS+CNN	0.3578	0.5624	0.4601	0.2965	0.2073	0.2519	0.3576	0.3243	0.3409	0.5725	0.5800	0.5762
JFSSL+CNN	0.4253	0.6654	0.5454	0.2878	0.1914	0.2396	0.3479	0.2971	0.3225	0.5726	0.5355	0.5540
Deep-SM	0.4120	0.6901	0.5511	0.3309	0.2668	0.2787	0.4986	0.4391	0.4688	0.6107	0.6003	0.6055
CMMN+CNN	**0.4380**	**0.6923**	**0.5652**	**0.4632**	**0.2840**	**0.3736**	**0.5787**	**0.4462**	**0.5124**	**0.7087**	**0.6096**	**0.6621**

TABLE III
MAP COMPARISON OF DIFFERENT CROSS-MODAL HASHING METHODS ON THE WIKI DATASET WITH 16,32,64 AND 128 BITS CODE LENGTH

	Image *vs.* Text				Text *vs.* Image				Average			
Method	16 bits	32 bits	64 bits	128 bits	16 bits	32 bits	64 bits	128 bits	16 bits	32 bits	64 bits	128 bits
CMSSH [29]	0.1877	0.1771	0.1646	0.1552	0.1630	0.1617	0.1539	0.1517	0.1754	0.1694	0.1593	0.1535
CVH [30]	0.1257	0.1212	0.1215	0.1171	0.1185	0.1034	0.1024	0.0990	0.1221	0.1123	0.1119	0.1081
IMH [31]	0.1573	0.1575	0.1568	0.1651	0.1463	0.1311	0.1290	0.1301	0.1518	0.1443	0.1429	0.1476
CMFH [33]	0.2132	0.2259	0.2362	0.2419	0.4884	0.5132	0.5269	0.5375	0.3508	0.3695	0.3816	0.3897
SCM [36]	0.2210	0.2337	0.2442	0.2596	0.2134	0.2366	0.2479	0.2573	0.2172	0.2351	0.2460	0.2584
SePH [10]	0.2787	0.2956	0.3064	0.3134	0.6318	0.6581	0.6646	0.6709	0.4553	0.4768	0.4855	0.4922
CMMN	0.2772	0.2831	0.3044	0.3097	0.6033	0.6197	0.6389	0.6451	0.4402	0.4514	0.4714	0.4774
CMSSH+CNN	0.1871	0.1987	0.2089	0.2069	0.1714	0.1664	0.1491	0.1483	0.1792	0.1825	0.1790	0.1776
CVH+CNN	0.1733	0.1723	0.1645	0.1608	0.2831	0.2076	0.1541	0.1441	0.2282	0.1900	0.1593	0.1525
IMH+CNN	0.2119	0.1883	0.1674	0.1586	0.3003	0.2651	0.2449	0.2317	0.2561	0.2267	0.2061	0.1951
CMFH+CNN	0.2053	0.2397	0.2395	0.2291	0.3299	0.3886	0.3738	0.3181	0.2676	0.3141	0.3066	0.2736
SCM+CNN	0.1807	0.1712	0.1698	0.1707	0.6695	0.6911	0.6833	**0.7002**	0.4251	0.4312	0.4265	0.4355
SePH+CNN	**0.4220**	**0.4507**	**0.4544**	**0.4561**	0.6254	0.6384	0.6413	0.6485	0.5237	0.5445	0.5478	0.5523
coAE+CNN	0.3659	0.3753	0.3744	0.3792	0.2360	0.2417	0.2276	0.2215	0.3009	0.3085	0.3010	0.3004
DSH+CNN	0.3109	0.3949	0.3863	0.3889	0.2866	0.4424	0.4942	0.5997	0.2988	0.4186	0.4402	0.4943
DCMH+CNN	0.3724	0.4366	0.4369	0.3521	0.6169	0.6610	0.6011	0.5635	0.4947	**0.5488**	0.5190	0.4578
CMMN+CNN	0.3953	0.4126	0.4390	0.4483	**0.6915**	**0.6742**	**0.6917**	0.6714	**0.5434**	0.5434	**0.5654**	**0.5598**

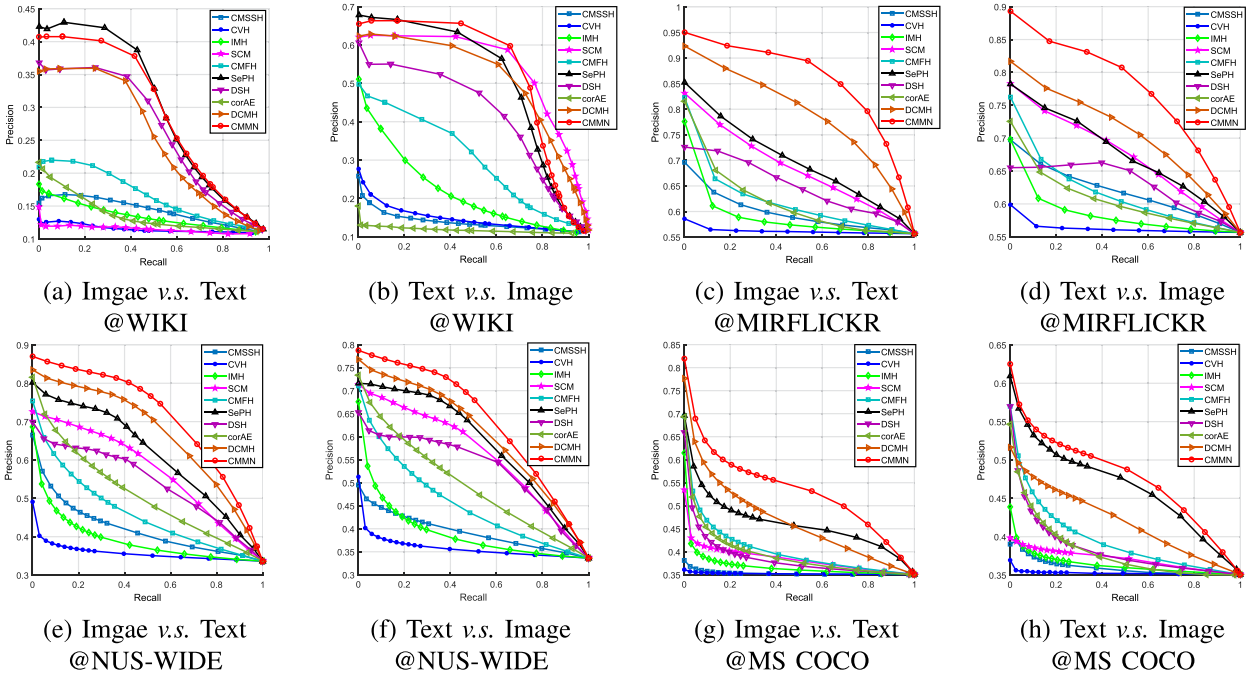
2) Results on Wiki: Table II reports our experimental mAP results on Wiki dataset over real-valued representation learning methods. Table III reports mAP results over hash-based methods using various numbers of bits. Note that, 'Method+CNN' denotes that the method uses the 4096-dimensional deep feature (extracted from the `fc7` layer of AlexNet pre-trained on ImageNet with Caffe [57]) as the image representation.

From Table II, we can see that our CMMN method yields great improvements (e.g., 17.2% over TV-CCA, 10.5% over LCFS and 1.98% over JFSSL) compared with other cross-modal real-valued representation learning baselines. And CMMN obtains comparable performance with Deep-SM since the semantic-preserving binary code.

From Table III we find that the CMMN method can outperform all the other baselines except the SePH method. Please note that SePH is a kernel-based method, which constructs kernels based on the image features and text features. The better performance of SePH may come from the kernel embedding feature which is more suitable for the small-scale dataset. However, based on the CNN visual feature, the CMMN+CNN improves the performance more than SePH+CNN and achieves the best performance (16 bits: 54.3%, 64 bits: 56.54% and 128 bits: 55.9%). The main reason for this is that the CMMN method can fully mine relevant facts of deep features to represent high-level semantic concepts by the memory mechanism. Besides, compared with deep models, CMMN outperforms the best DCMH method with a large margin on average.

Fig. 4(a) and (b) show the Precision-Recall curves of compared methods with 64 bits code. CMMN method performs comparably with the state-of-the-art baselines, which is consistent with the result in Table III.

3) Results on MIRFLICKR: Table II reports the NDCG result over real-valued method. Table IV reports the NDCG result over hash-based methods using various numbers of bits. Fig. 4(c) and (d) show the Precision-Recall curves.



(a) Imgae *v.s.* Text @WIKI (b) Text *v.s.* Image @WIKI (c) Imgae *v.s.* Text @MIRFLICKR (d) Text *v.s.* Image @MIRFLICKR

(e) Imgae *v.s.* Text @NUS-WIDE (f) Text *v.s.* Image @NUS-WIDE (g) Imgae *v.s.* Text @MS COCO (h) Text *v.s.* Image @MS COCO

Fig. 4. Precision-recall curves on Wiki, MS COCO, Mirflickr and NUS-WIDE datasets. The code length is 64.

TABLE IV
NDCG@500 OF DIFFERENT CROSS-MODAL HASHING METHODS WITH 16,32,64,128 BITS ON MIFLICKR DATASET

	Imgae *v.s.* Text				Text *v.s.* Image				Average			
Method	16bits	32bits	64bits	128 bits	16bits	32bits	64bits	128 bits	16bits	32bits	64bits	128 bits
CMSSH [29]	0.2488	0.2634	0.2460	0.2223	0.2219	0.2127	0.2479	0.2103	0.2354	0.2381	0.2470	0.2163
CVH [30]	0.1884	0.1851	0.1907	0.1931	0.1917	0.1917	0.1941	0.2006	0.1900	0.1884	0.1924	0.1968
IMH [31]	0.2937	0.2772	0.2616	0.2458	0.2755	0.2647	0.2513	0.2377	0.2846	0.2710	0.2565	0.24175
SCM [36]	0.3229	0.3449	0.3573	0.3628	0.2959	0.3105	0.3222	0.3256	0.3094	0.3277	0.3397	0.3442
CMFH [33]	0.2908	0.3059	0.3099	0.3162	0.2830	0.3012	0.3054	0.3054	0.2869	0.3035	0.3076	0.3108
SePH [10]	0.4216	0.4416	0.4506	0.4749	0.3089	0.3260	0.3136	0.3563	0.3652	0.3838	0.3821	0.4156
DSH [13]	0.2653	0.2603	0.2533	0.2545	0.2424	0.2465	0.2341	0.2206	0.2539	0.2534	0.2437	0.2375
corAE [26]	0.3221	0.4044	0.4591	0.4599	0.2831	0.2885	0.3268	0.2987	0.3026	0.3464	0.3930	0.3793
DCMH [16]	0.4064	0.4305	0.4553	0.4623	0.3132	0.3348	0.3392	0.3367	0.3598	0.3826	0.3972	0.3995
CMMN	**0.5054**	**0.5378**	**0.5640**	**0.5787**	**0.4020**	**0.4275**	**0.4381**	**0.4462**	**0.4537**	**0.4834**	**0.5010**	**0.5124**

Our CMMN method dramatically outperforms other hashing methods, as listed in Table IV. Specifically, the NDCG values of the proposed method indicate 8.8% $\sim$ 11.8% relative increase over the best baseline SePH. Besides, CMMN outperforms the state-of-the-art DCMH method by a large margin (9.3% of 16 bits, 10% of 32 bits, 10.3% of 64 bits and 11.2% of 128 bits) since DCMH only considering the binary similarity (similar or dissimilar) and ignoring the complex similarity relationship of data with multi-labels. The result validates the advantage of CMMN method for multi-label cross-modal retrieval task.

4) Results on NUS-WIDE: Table II reports the NDCG result over real-valued method on NUS-WIDE dataset. Table V report the NDCG result over hash-based methods. And Fig. 4(e) and (f) show the Precision-Recall curves.

Like MIRFLICKR, the proposed CMMN method gains all-around advantages over all the other cross-modal hashing methods on NUS-WIDE, as can be seen in Table Table V. To be more specific, the NDCG values of the proposed method indicate a 4.7% $\sim$ 6.8% relative increase over the second best baseline SePH. Besides, the CMMN method maintains its advantage over the best baseline DCMH. The result confirms the superior of CMMN method over other traditional deep cross-modal methods.

5) Results on Microsoft COCO: Table VI reports the mAP result, Table II and VII report the NDCG result, and Table VIII reports the Med r result over hash-based methods. Fig. 4(g) and (h) show the Precision-Recall curves. We can see that the CMMN method outperforms other baselines under all used evaluation metrics. Compared with non-deep methods, the CMMN method yields great improvements over the best performing non-deep baseline SePH concerning mAP value and obtains comparable performance on NDCG value.

Besides, compared with deep models, our CMMN method outperforms the state-of-the-art deep cross-modal hash method DCMH with a large margin (10.4% of 16 bits, 9.6% of 32 bits, 8.8% of 64 bits, and 6.3% of 128 bits) on NDCG value. The reason may be explained as follows. DCMH only considers to learn cross-modal similarity but ignores the intra-modal similarity. Also, DCMH prefers to preserve the simple binary similarity (similar or dissimilar) rather than the complex

TABLE V
NDCG@500 OF DIFFERENT CROSS-MODAL HASHING METHODS WITH 16,32,64,128 BITS ON NUS-WIDE DATASET

	Imgae *v.s.* Text				Text *v.s.* Image				Average			
Method	16bits	32bits	64bits	128 bits	16bits	32bits	64bits	128 bits	16bits	32bits	64bits	128 bits
CMSSH [29]	0.4228	0.4092	0.4391	0.4485	0.2955	0.3333	0.3305	0.3199	0.3591	0.3712	0.3848	0.3842
CVH [30]	0.2843	0.2928	0.2911	0.2928	0.2892	0.2965	0.2951	0.2938	0.2868	0.2946	0.2931	0.2933
IMH [31]	0.4450	0.4593	0.4448	0.4084	0.4533	0.4602	0.4440	0.4103	0.4491	0.4597	0.4444	0.4093
SCM [36]	0.5075	0.5149	0.5299	0.5308	0.4941	0.5010	0.5141	0.5143	0.5008	0.5080	0.5220	0.5226
CMFH [33]	0.4875	0.5012	0.5270	0.5394	0.4642	0.4775	0.4998	0.5091	0.4758	0.4893	0.5134	0.5242
SePH [10]	0.6157	0.6251	0.6335	0.6493	0.5275	0.5320	0.5251	0.5353	0.5716	0.5786	0.5793	0.5923
DSH [13]	0.4430	0.4516	0.4769	0.4685	0.4613	0.4167	0.4379	0.4463	0.4521	0.4341	0.4574	0.4574
corAE [26]	0.4095	0.4382	0.5148	0.4961	0.3904	0.4369	0.5234	0.4536	0.3999	0.4375	0.5191	0.4748
DCMH [16]	0.5757	0.6159	0.6079	0.6237	0.5756	0.5858	0.5901	0.6007	0.5756	0.6008	0.5990	0.6122
CMMN	**0.6450**	**0.6645**	**0.7002**	**0.7087**	**0.5762**	**0.5890**	**0.6096**	**0.6154**	**0.6106**	**0.6267**	**0.6549**	**0.6621**

TABLE VI
MAP@500 OF DIFFERENT CROSS-MODAL HASHING METHODS WITH 16,32,64,128 BITS ON MICROSOFT COCO DATASET

	Imgae *v.s.* Text				Text *v.s.* Image				Average			
Method	16bits	32bits	64bits	128 bits	16bits	32bits	64bits	128 bits	16bits	32bits	64bits	128 bits
CMSSH [29]	0.3941	0.4029	0.3864	0.3912	0.3794	0.4090	0.4159	0.3933	0.3868	0.4059	0.4012	0.3922
CVH [30]	0.3706	0.3658	0.3698	0.3730	0.3672	0.3658	0.3698	0.3733	0.3689	0.3658	0.3698	0.3731
IMH [31]	0.5196	0.5562	0.5482	0.5130	0.4410	0.4407	0.4326	0.4259	0.4803	0.4985	0.4904	0.4695
SCM [36]	0.4615	0.4858	0.4738	0.4162	0.3945	0.3988	0.3989	0.3821	0.4280	0.4423	0.4364	0.3992
CMFH [33]	0.5605	0.6088	0.6530	0.6809	0.5216	0.5319	0.5585	0.5775	0.5411	0.5704	0.6058	0.6292
SePH [10]	0.6221	0.6438	0.6588	0.6721	0.5398	0.5480	0.5548	0.5645	0.5809	0.5959	0.6068	0.6183
DSH [13]	0.5199	0.5886	0.6217	0.6304	0.4557	0.5097	0.5529	0.5644	0.4878	0.5492	0.5873	0.5974
corAE [26]	0.5424	0.6120	0.6502	0.6574	0.4559	0.5293	0.5348	0.5384	0.4991	0.5706	0.5925	0.5979
DCMH [16]	0.6254	0.6958	0.7370	0.7756	0.5098	0.5058	0.5144	0.5910	0.5676	0.6008	0.6257	0.6833
CMMN	**0.7582**	**0.7834**	**0.7910**	**0.7795**	**0.5762**	**0.6030**	**0.6166**	**0.6032**	**0.6672**	**0.6932**	**0.7038**	**0.6913**

TABLE VII
NDCG@500 OF DIFFERENT CROSS-MODAL HASHING METHODS WITH 16,32,64,128 BITS ON MICROSOFT COCO DATASET

	Imgae *v.s.* Text				Text *v.s.* Image				Average			
Method	16bits	32bits	64bits	128 bits	16bits	32bits	64bits	128 bits	16bits	32bits	64bits	128 bits
CMSSH [29]	0.1120	0.1117	0.1058	0.1012	0.1020	0.1091	0.1116	0.1063	0.1070	0.1104	0.1087	0.1038
CVH [30]	0.1004	0.0983	0.1005	0.1024	0.1002	0.0991	0.1012	0.1033	0.1003	0.0987	0.1008	0.1028
IMH [31]	0.1931	0.2077	0.1936	0.1690	0.1487	0.1522	0.1455	0.1370	0.1709	0.1799	0.1695	0.1530
SCM [36]	0.1494	0.1735	0.1610	0.1270	0.1194	0.1288	0.1260	0.1126	0.1344	0.1512	0.1435	0.1198
CMFH [33]	0.2224	0.2571	0.2899	0.3098	0.1982	0.2182	0.2398	0.2539	0.2103	0.2377	0.2649	0.2819
SePH [10]	0.3619	0.4138	0.4376	0.4456	0.2214	0.2413	0.2732	0.2832	0.2917	0.3276	0.3554	0.3644
DSH [13]	0.1943	0.2488	0.2720	0.2730	0.1621	0.2036	0.2302	0.2374	0.1782	0.2262	0.2511	0.2552
corAE [26]	0.2094	0.2636	0.2900	0.2912	0.1658	0.2126	0.2267	0.2276	0.1876	0.2381	0.2583	0.2594
DCMH [16]	0.2569	0.3191	0.3594	0.3979	0.1624	0.1803	0.1946	0.2227	0.2097	0.2497	0.2770	0.3103
CMMN	**0.3921**	**0.4281**	**0.4576**	**0.4632**	**0.2372**	**0.2634**	**0.2738**	**0.2840**	**0.3146**	**0.3458**	**0.3657**	**0.3736**

TABLE VIII
MED R RESULT OF DIFFERENT METHODS WITH 16,32,64,128 BITS ON MICROSOFT COCO DATASET. MED R IS THE MEDIAN RANK OF THE FIRST CLOSET GROUND TRUTH (LOW IS GOOD)

	Imgae *v.s.* Text				Text *v.s.* Image			
Method	16	32	64	128	16	32	64	128
CMSSH	703	766	867	927	1519	1681	2065	2023
CVH	819	860	738	699	829	839	796	755
IMH	260	207	162	173	1571	1324	981	792
SCM	586	485	335	323	1475	1728	1851	2156
CMFH	167	123	81	72	870	789	697	548
SePH	241	173	125	104	1248	1336	1267	1328
DSH	271	178	116	94	1263	724	648	456
corAE	186	115	92	72	1174	935	912	846
DCMH	154	78	57	49	1402	1690	1358	1757
CMMN	**84**	**61**	**52**	**45**	**663**	**605**	**513**	**437**

similarity of multi-labels. While the CMMN method jointly utilizes semantic information and similarity information to learn semantic similarity-preserving codes. Therefore, the CMMN method works better than DCMH.

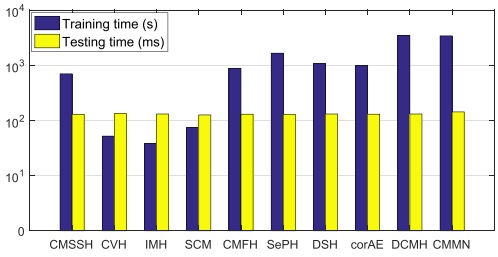


Fig. 5. Computational time of different methods with 128 bits code on the Microsoft COCO dataset.

Furthermore, the CMMN method achieves the best performance on Med r value. This result indicates that the CMMN method is prone to retrieve more relevant and accurate samples than other methods.

Fig. 6 shows example *Image vs. Text* search results and Fig. 7 shows example *Text vs. Image* search results for compared

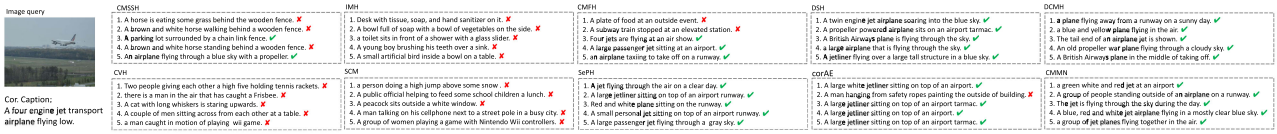


Fig. 6. Retrieval examples of *Image vs. Text* on Microsoft COCO with 32 bit. Top 5 results for each methods are shown. '×' denotes irrelevant.

TABLE IX
MAP COMPARISON OF DIFFERENT METHODS ON CMPLACES DATASET. EACH COLUMN SHOWS A DIFFERENT QUERY-TARGET PAIR. ON THE FAR RIGHT, WE AVERAGE OVER ALL PAIRS

Query	NAT				CLP				SPT				LDR				DSC				mea:
Target	CLP	SPT	LDR	DSC	NAT	SPT	LDR	DSC	NAT	CLP	LDR	DSC	NAT	CLP	SPT	DSC	NAT	CLP	SPT	LDR	mAP
BI [20]	17.8	15.5	10.1	0.8	11.4	13.1	9.0	0.8	9.0	10.1	5.6	0.8	4.9	7.6	6.8	0.8	0.6	0.9	0.9	0.9	6.4
Deep_A [20]	14.0	29.8	6.2	18.4	9.2	17.6	3.7	12.9	21.8	15.9	6.2	27.7	3.7	3.1	6.6	5.4	5.2	3.5	10.5	2.1	11.2
Deep_B [20]	17.8	23.7	9.5	5.6	13.4	18.1	8.9	4.6	16.7	16.2	8.8	5.3	6.2	8.1	9.4	3.3	3.0	4.1	4.6	2.8	9.5
Deep_C [20]	14.3	**32.1**	5.4	22.1	10.0	**19.1**	3.8	14.4	24.4	17.5	5.8	32.7	3.3	3.4	6.0	4.9	**15.1**	12.5	**32.6**	4.6	14.2
SePH [10]	0.9	12.1	25.6	6.8	1.2	0.7	0.8	0.6	**33.1**	0.8	22.0	5.5	**20.4**	0.8	6.3	3.7	12.8	0.8	3.7	7.8	8.32
CMMN	36.6	14.7	39.3	39.5	15.8	15.1	40.0	40.1	15.8	41.3	43.5	45.4	8.1	20.8	6.5	20.3	3.8	11.4	3.5	10.6	23.6
Deep-SM [4]	**45.1**	27.8	**47.0**	**51.1**	**21.3**	**21.7**	38.5	40.1	23.6	41.2	46.3	45.7	11.7	21.3	**12.0**	24.4	8.2	**14.5**	9.2	13.8	28.2
CMMN_SM	38.0	19.6	43.2	43.2	15.6	18.6	**46.7**	**46.2**	18.2	**52.7**	**57.7**	**58.3**	11.3	**25.8**	11.0	**28.6**	7.1	13.8	5.8	**14.1**	**28.8**

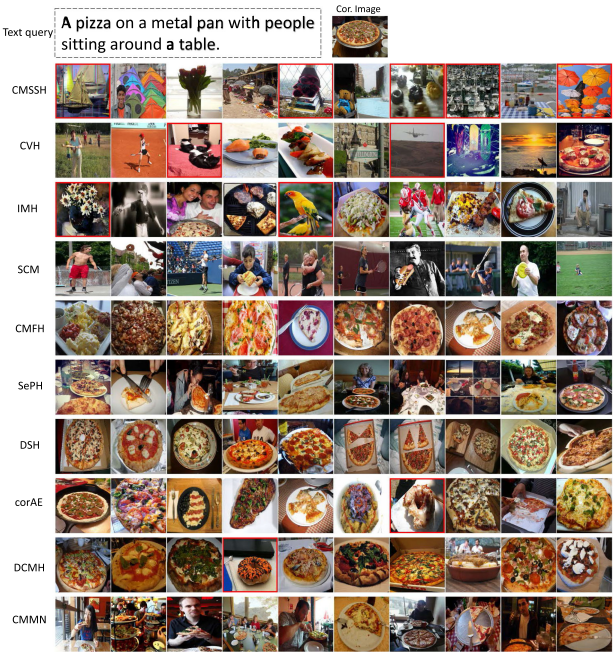


Fig. 7. Retrieval examples of *Text vs. Image* on Microsoft COCO with 32 bit. Top 10 results are shown. Results with 'red' border denote irrelevant.

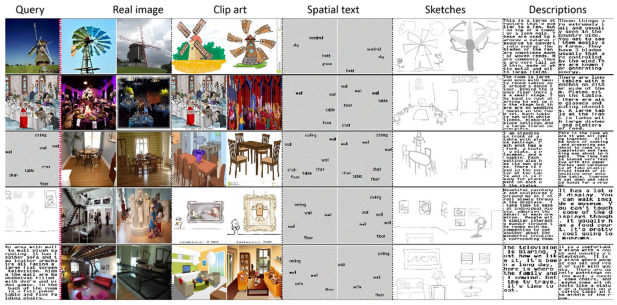


Fig. 8. Quality retrieval examples on CMPlaces. The first column represents the query, and top 2 results for each modality are shown.

methods. As can be seen from the latter figure, our CMMN method tends to retrieve more relevant images than the other baselines for the query containing certain concepts, i.e., "pizza," "pan," "person," and "table."

To discuss the difference of time cost between the CMMN method and other contrast methods, we report their training and test time in Fig. 5. Our PC is configured with a 3.20 GHz CPU, a TITAN X GPU and 32.0 GB RAM. All experiments are accelerated with GPU. Because all methods use the same feature, the training time only corresponds to the optimization. The test time consists of encoding and retrieval time. We can see that CMMN takes more training time than other deep baselines due to the additional operations of the memory. However, owing to the fast coding of the CMMN method, the test time of CMMN is comparable to others.

6) Results on CMPlaces: Table IX reports the mAP results on CMPlaces dataset, where each "Query-Target" pair means using the testing data of "Query" modality to retrieve relevant data of "Target" modality. As this dataset is extremely noisy and diverse, the absolute mAP for all methods is low. We can see that the CMMN method outperforms the best configuration Deep_C [20] with a large margin 9.4%. The superior performance of CMMN method demonstrates its effectiveness for unaligned cross-modal data.

After further observation, we see that, for most cross-modalities retrieval tasks, significant improvements could be achieved with our CMMN method(e.g., for "NAT-CLP," from 14.3% obtained by Deep_C to 36.6%). However, for several cross-modal retrieval tasks, the results of the CMMN method do not show the consistent improvement, e.g., for "DSC-NAT," the CMMN achieves 3.8%, and Deep_C achieves 15.1%. To explain this, we analyze the learned testing CMMN feature of each modality and find that the discriminative power of DSC is poorer (with a classification accuracy of 4.5%) than that of other modalities, which results in the unexpected performance.

Besides, since the CMPlace dataset is noisy and the memory data of CMMN is low discriminative, our CMMN method may not take advantage of the memory mechanism to boost perfor-

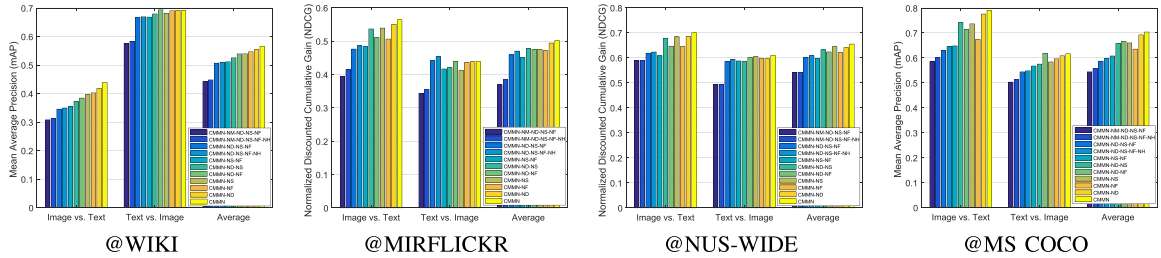


Fig. 9. Evaluations (mAP and NDCG) of the proposed CMMN model with ablating different components. The modified CMMN method is called 'CMMN-*.' '-NM' denotes no memory block. '-ND' denotes no modality classifier D (for adversarial learning). '-NS' denotes no pairwise similarity loss term. '-NF' denotes no fusion of aligned multi-modal features. '-NH' denotes no hashing quantization.

mance and is inferior to the Deep-SM method. While based on the semantic-matching, CMMN_{SM} achieves comparable cross-modal retrieval performance 28.8% with Deep-SM (28.2%). This result shows that the learned semantic representations of CMMN and Deep-SM are consistent.

Fig. 8 shows the quality retrieval examples of different modalities queries. As can be seen, our CMMN method can return relevant results for queries containing the specific semantic concept, e.g., "windmill," "table" and "sofa." It confirms that our CMMN method can effectively capture the similarities of unaligned cross-modal data.

To sum up, based on the above reported experimental results, the CMMN is effective for cross-modal retrieval.

E. Discussion

In this section, we further analyze the reason for the superior performance of the proposed CMMN method. Firstly, we investigate the effectiveness of memory contents initialization approach in Eq.(1). Secondly, we separately remove important components and steps to evaluate their influence on the final performance. These components and steps include: 1) memory block for representation learning, 2) modality classifier D for fine-fusing multi-modal features, 3) similarity-preserving loss term, 4) fusion of aligned multi-modal codes, 5) quantization of hashing. The modified CMMN method is called 'CMMN-*.' Here '-NM' denotes no memory block. '-ND' denotes no modality classifier D. '-NS' denotes no $\text{J}_{\text{similarity}}$ loss term. '-NF' denotes no fusion of aligned multi-modal features. '-NH' denotes no hashing quantization. Thirdly, we visualize the distribution of the learned CMMN feature to inspect its property of semantic similarity-preserving. Finally, we analyze the influence of memory block component and hash quantization for the time performance. We conduct experiments on Wiki, Microsoft COCO, MIFLICKR and NUS-WIDE datasets. The model parameters and training parameters are set according to Section IV-B and IV-C, code length is 64.

Impact of memory initialization method: Here, we empirically compare the memory initialization approach with three simple methods. $\text{CMMN}_{\text{zeroMem}}$ initializes the memory with zero. $\text{CMMN}_{\text{randWMem}}$ initializes the memory with the random value from Gaussian distribution, $\text{CMMN}_{\text{randSMem}}$ initializes the memory by sampling data. Tabel X reports the result. As we can see our method achieves the best performance on all datasets, which validates the effectiveness and correctness of our memory initialization approach.

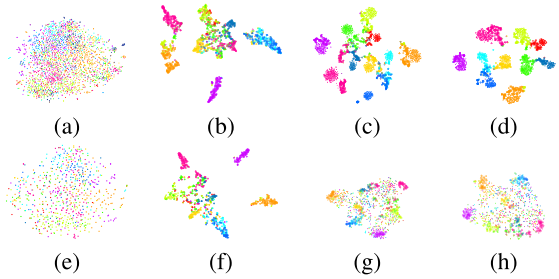


Fig. 10. A visualization of training and testing data. Different colors indicate different categories. Dot '·' and circle '○' indicate image and text respectively. The first and second rows represent training data and testing data respectively. (a) and (e) illustrate the distribution of CNN visual representation. (b) and (f) show the distribution of text LDA descriptor. (c), (d), (g), and (h) show the distribution of feature learned from CMMN, which is trained either without modality classifier D in (c) and (g) or with modality classifier D (for adversarial learning) in (d) and (h).

Evaluation of different components: Fig. 9 shows the results. We can see that the CMMN-ND-NS-NF outperforms the CMMN-NM-ND-NS-NF that missing memory component with the same configuration. This result demonstrates the importance of memory for cross-modal representation learning.

Also, the CMMN-ND-NS-NF method obtains a comparable performance with CMMN-ND-NS-NF-NH. It shows that the output of the hash layer is saturated, the quantization of hashing will not hurt the performance.

Besides, separately removing the modality classifier D, the pairwise similarity loss term, and the fusion of aligned multi-modal codes will damage the retrieval performance of CMMN method to varying degrees, e.g., the performance of CMMN-ND is inferior to CMMN. Indeed, the learned common features of CMMN belonging to the same category appear to be more compact than that of the CMMN-ND method (as shown in Fig. 10(c) and (d)). Therefore, we can conclude that different components of CMMN are essential for cross-modal representation learning and have separated contributions to the final retrieval performance.

However, we may note that the CMMN method achieves a better performance for *Text vs. image* task than that for *Image query vs. Text* task on Wiki dataset. To explain this observation, Table XI gives the uni-modal classification accuracy (3-layers MLP with 4,096 hidden nodes is utilized as the classifier) and retrieval performance (mAP) of different modalities features (i.e., LDA, CNN, and their corresponding CMMN features CMMN_{LDA} and CMMN_{CNN}). We observe that the

TABLE X
THE COMPARISON (MAP AND NDCG) OF DIFFERNET MEMORY INITIALIZATION METHODS ON FOUR DATASETS

	Wiki (mAP)			Microsoft COCO (NDCG)			MIFLICKR (NDCG)			NUS-WIDE (NDCG)		
Method	I *vs.* T	T *vs.* I	Avg	I *vs.* T	T *vs.* I	Avg	I *vs.* T	T *vs.* I	Avg	I *vs.* T	T *vs.* I	Avg
$\text{CMMN}_{\text{zeroMem}}$	0.3535	0.6301	0.4918	0.4149	0.2487	0.3318	0.5141	0.3726	0.4434	0.6764	0.5639	0.6021
$\text{CMMN}_{\text{randWMem}}$	0.3363	0.6265	0.4814	0.4193	0.2447	0.3320	0.5158	0.3662	0.4410	0.6770	0.5578	0.6174
$\text{CMMN}_{\text{randSMem}}$	0.3336	0.6373	0.4854	0.4092	0.2403	0.3247	0.5232	0.3319	0.4275	0.6802	0.5583	0.6193
CMMN	**0.4390**	**0.6917**	**0.5654**	**0.4576**	**0.2738**	**0.3657**	**0.5640**	**0.4381**	**0.5010**	**0.7002**	**0.6096**	**0.6549**

TABLE XI
CLASSIFICATION ACCURACY AND UNI-MODALITY RETRIEVAL PERFORMANCE (MAP) ON THE WIKI WITH DIFFERENT FEATURES

Feature	Dim	Train accuracy	Test accuracy	Retrieval
CNN	4096	0.4552	0.3969	0.1801
LDA	10	0.6759	0.6452	0.5391
CMMN_{CNN}	64	**0.9256**	0.4236	0.3939
CMMN_{LDA}	64	0.8678	**0.6631**	**0.6235**

TABLE XII
TIME COST OF THE PROPOSED CMMN METHOD (64 BITS) WITH DIFFERENT CONFIGURATIONS ON MICROSOFT COCO DATASET

	Training (100s)		Testing (ms)	
Method	MLPs	CMMN	Encode	Retrieval
CMMN-NM	8.671	5.808	2.072	164.1
CMMN-NM-NH			2.071	410.4
CMMN	8.671	42.30	2.447	161.7
CMMN-NH			2.446	405.3

CMMN_{LDA} feature possesses a greater discriminative ability than the CMMN_{CNN} feature, which may result in the performance gap of the CMMN method.

Quality of learned feature: We use t-SNE tools to embed high-dimensional features of Wiki dataset (i.e., LDA, CNN and their corresponding CMMN features of CMMN-ND-NS-NF and CMMN-NS-NF methods) into 2-dimension space and visualize their distribution in Fig. 10. As can be seen from the Fig. 10(c), after mapped by CMMN, both text LDA and image CNN features provide a better separation between different categories. Also, the common representations belonging to the same category from different modalities appear to be compact, which indicates that the CMMN method simultaneously preserves the semantic similarity of intra-modal and inter-modal. Moreover, when we compare Fig. 10(c) and Fig. 10(d), we can find that the common representations of different modalities in Fig. 10(d) are more compact than that in Fig. 10(c). This result confirms that the adversarial learning strategy effectively discards the influence of modality in the CMMN feature.

However, different from Fig. 10(c), the common representations of testing images are scattered in the embedding space as shown in Fig. 10(g). One possible reason is that most semantic concepts of Wiki dataset are abstract (e.g., history and art). Therefore, the image content of the same category is quite different and using real object-level memory contents cannot fully represent the complex abstract semantic concepts of an image.

Time cost: Table XII reports the time cost of CMMN-NM, CMMN-NH, and CMMN methods on Microsoft COCO dataset. The training time includes the cost of training MLPs for preprocessing and optimizing CMMN network. The testing time is composed of encoding time and retrieval time. As can be seen from Table XII, CMMN takes more time than CMMN-NM in training since the introduction of memory block causes additional computations. In the testing stage, CMMN performs faster retrieval than CMMN-NH via hash coding, which confirms the efficiency of the hash layer.

To sum up, for the CMMN method, the memory block component is helpful for cross-modal representation learning but brings extra time cost. Both the fine-fusion with adversarial learning and the fusion of aligned codes can eliminate the modality information of learned feature. The semantic hash layer can output semantic similarity-preserving binary codes for accurate and efficient cross-modal retrieval.

V. CONCLUSION

In this paper, we proposed a memory network termed CMMN for cross-modal retrieval. Unlike existing cross-modal methods that learn projection over raw features directly, our CMMN exploits memory mechanism to pre-store the discriminative features of available modalities in memories as potentially relevant components which are used to re-express the modal-specific feature. When an input feature of a special modality comes, CMMN can find supporting facts from memories with the soft-attention mechanism and learn common representation through aggregating and transforming these features. Experimental results on three datasets have demonstrated the effectiveness of the proposed approach. For future work, it is interesting to extend our approach to cross-dataset retrieval, e.g., exploiting available knowledge among different databases for general large-scale search.

REFERENCES

[1] K. Wang, Q. Yin, W. Wang, S. Wu, and L. Wang, "A comprehensive survey on cross-modal retrieval," 2016, arXiv: 1607.06215.
[2] K. Wang, R. He, L. Wang, W. Wang, and T. Tan, "Joint feature selection and subspace learning for cross-modal retrieval," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 38, no. 10, pp. 2010–2023, Oct. 2016.
[3] A. Salvador *et al.*, "Learning cross-modal embeddings for cooking recipes and food images," in *Proc. IEEE Conf. Comput. Vision Pattern Recognit.*, 2017, pp. 3068–3076.
[4] Y. Wei *et al.*, "Cross-modal retrieval with CNN visual features: A new baseline," *IEEE Trans. Cybern.*, vol. 47, no. 2, pp. 449–460, Feb. 2017.
[5] Y. Gong, Q. Ke, M. Isard, and S. Lazebnik, "A multi-view embedding space for modeling internet images, tags, and their semantics," *Int. J. Comput. Vision*, vol. 106, no. 2, pp. 210–233, 2014.
[6] L. Zhang, B. Ma, G. Li, Q. Huang, and Q. Tian, "Generalized semi-supervised and structured subspace learning for cross-modal retrieval," *IEEE Trans. Multimedia*, vol. 20, no. 1, pp. 128–141, Jan. 2018.
[7] N. Rasiwasia *et al.*, "A new approach to cross-modal multimedia retrieval," in *Proc. 18th ACM Int. Conf. Multimedia*, 2010, pp. 251–260.
[8] G. Andrew, R. Arora, J. A. Bilmes, and K. Livescu, "Deep canonical correlation analysis," in *Proc. 30th Int. Conf. Mach. Learn.*, 2013, pp. 1247–1255.
[9] Y. Cao, M. Long, J. Wang, Q. Yang, and P. S. Yu, "Deep visual-semantic hashing for cross-modal retrieval," in *Proc. 22nd ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, 2016, pp. 1445–1454.

[10] Z. Lin, G. Ding, M. Hu, and J. Wang, "Semantics-preserving hashing for cross-view retrieval," in *Proc. IEEE Conf. Comput. Vision Pattern Recognit.*, 2015, pp. 3864–3872.
[11] Y. Peng, J. Qi, X. Huang, and Y. Yuan, "CCL: Cross-modal correlation learning with multi-grained fusion by hierarchical network," *IEEE Trans. Multimedia*, vol. 20, no. 2, pp. 405–420, Feb. 2018.
[12] R. Rosipal and N. Krämer, "Overview and recent advances in partial least squares," in *Proc. Int. Conf. Subspace, Latent Struct. Feature Selection*, 2005, pp. 34–51.
[13] H. Liu, R. Wang, S. Shan, and X. Chen, "Deep supervised hashing for fast image retrieval," in *Proc. IEEE Conf. Comput. Vision Pattern Recognit.*, 2016, pp. 2064–2072.
[14] K. Lin, H. Yang, J. Hsiao, and C. Chen, "Deep learning of binary hash codes for fast image retrieval," in *Proc. IEEE Conf. Comput. Vision Pattern Recognit. Workshops*, 2015, pp. 27–35.
[15] H. Liu, R. Ji, Y. Wu, F. Huang, and B. Zhang, "Cross-modality binary code learning via fusion similarity hashing," in *Proc. IEEE Conf. Comput. Vision Pattern Recognit.*, 2017, pp. 6345–6353.
[16] Q. Jiang and W. Li, "Deep cross-modal hashing," in *Proc. IEEE Conf. Comput. Vision Pattern Recognit.*, 2017, pp. 3270–3278.
[17] M. J. Huiskes and M. S. Lew, "The MIR flickr retrieval evaluation," in *Proc. 1st ACM SIGMM Int. Conf. Multimedia Inf. Retrieval*, Vancouver, BC, Canada, Oct. 30–31, 2008, pp. 39–43.
[18] T.-S. Chua *et al.*, "NUS-WIDE: A real-world web image database from national university of singapore," in *Proc. ACM Int. Conf. Image Video Retrieval*, Jul. 8–10, 2009, Art. no. 48.
[19] T.-Y. Lin *et al.*, "Microsoft COCO: Common objects in context," in *Proc. Eur. Conf. Comput. Vision*, 2014, pp. 740–755.
[20] L. Castrejon, Y. Aytar, C. Vondrick, H. Pirsiavash, and A. Torralba, "Learning aligned cross-modal representations from weakly aligned data," in *Proc. IEEE Conf. Comput. Vision Pattern Recognit.*, 2016, pp. 2940–2949.
[21] G. Song and X. Tan, "Cross-modal retrieval via memory network," in *Proc. Brit. Mach. Vision Conf.*, 2017.
[22] A. Sharma, A. Kumar, H. Daume, and D. W. Jacobs, "Generalized multiview analysis: A discriminative latent space," in *Proc. IEEE Conf. Comput. Vision Pattern Recognit.*, 2012, pp. 2160–2167.
[23] X. Jiang *et al.*, "Deep compositional cross-modal learning to rank via local–global alignment," in *Proc. 23rd ACM Int. Conf. Multimedia*, 2015, pp. 69–78.
[24] N. Srivastava and R. Salakhutdinov, "Multimodal learning with deep Boltzmann machines," *J. Mach. Learn. Res.*, vol. 15, no. 1, pp. 2949–2980, 2014.
[25] J. Ngiam *et al.*, "Multimodal deep learning," in *Proc. 28th Int. Conf. Mach. Learn.*, 2011, pp. 689–696.
[26] F. Feng, X. Wang, and R. Li, "Cross-modal retrieval with correspondence autoencoder," in *Proc. 22nd ACM Int. Conf. Multimedia*, 2014, pp. 7–16.
[27] L. Lin, G. Wang, W. Zuo, X. Feng, and L. Zhang, "Cross-domain visual matching via generalized similarity measure and feature learning," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 39, no. 6, pp. 1089–1102, Jun. 2017.
[28] V. E. Liong, J. Lu, Y. Tan, and J. Zhou, "Deep coupled metric learning for cross-modal matching," *IEEE Trans. Multimedia*, vol. 19, no. 6, pp. 1234–1244, Jun. 2017.
[29] D. Wang, G. Song, and X. Tan, "Bayesian denoising hashing for robust image retrieval," *Pattern Recognit.*, vol. 86, pp. 134–142, 2019.
[30] G. Song and X. Tan, "Learning multilevel semantic similarity for large-scale multi-label image retrieval," in *Proc. ACM Int. Conf. Multimedia Retrieval*, 2018, pp. 64–72.
[31] M. M. Bronstein, A. M. Bronstein, F. Michel, and N. Paragios, "Data fusion through cross-modality metric learning using similarity-sensitive hashing," in *Proc. IEEE Comput. Soc. Conf. Comput. Vision Pattern Recognit.*, 2010, pp. 3594–3601.
[32] S. Kumar and R. Udupa, "Learning hash functions for cross-view similarity search," in *Proc. 22nd Int. Joint Conf. Artif. Intell.*, 2011, pp. 1360–1365.
[33] J. Song, Y. Yang, Y. Yang, Z. Huang, and H. T. Shen, "Inter-media hashing for large-scale retrieval from heterogeneous data sources," in *Proc. ACM SIGMOD Int. Conf. Manage. Data*, 2013, pp. 785–796.
[34] J. Zhou, G. Ding, and Y. Guo, "Latent semantic sparse hashing for cross-modal similarity search," in *Proc. 37th Int. ACM SIGIR Conf. Res. Develop. Inf. Retrieval*, 2014, pp. 415–424.
[35] G. Ding, Y. Guo, and J. Zhou, "Collective matrix factorization hashing for multimodal data," in *Proc. IEEE Conf. Comput. Vision Pattern Recognit.*, 2014, pp. 2083–2090.
[36] Y. Zhen, P. Rai, H. Zha, and L. Carin, "Cross-modal similarity learning via pairs, preferences, and active supervision," in *Proc. 29th AAAI Conf. Artif. Intell.*, 2015, pp. 3203–3209.
[37] J. Zhou, G. Ding, Y. Guo, Q. Liu, and X. Dong, "Kernel-based supervised hashing for cross-view similarity search," in *Proc. IEEE Int. Conf. Multimedia Expo*, 2014, pp. 1–6.
[38] D. Zhang and W. Li, "Large-scale supervised multimodal hashing with semantic correlation maximization," in *Proc. 28th AAAI Conf. Artif. Intell.*, 2014, pp. 2177–2183.
[39] L. Liu, F. Shen, Y. Shen, X. Liu, and L. Shao, "Deep sketch hashing: Fast free-hand sketch-based image retrieval," in *Proc. IEEE Conf. Comput. Vision Pattern Recognit.*, 2017, pp. 2298–2307.
[40] J. Weston, S. Chopra, and A. Bordes, "Memory networks," 2014, arXiv:1410.3916.
[41] S. Sukhbaatar, A. Szlam, J. Weston, and R. Fergus, "End-to-end memory networks," in *Proc. 28th Int. Conf. Neural Inf. Process. Syst.*, 2015, pp. 2440–2448.
[42] A. H. Miller *et al.*, "Key-value memory networks for directly reading documents," in *Proc. Conf. Empirical Methods Natural Lang. Process.*, 2016, pp. 1400–1409.
[43] A. Graves, G. Wayne, and I. Danihelka, "Neural turing machines," arXiv preprint arXiv:1410.5401, 2014.
[44] Y. Kim, C. Denton, L. Hoang, and A. M. Rush, "Structured attention networks," in *Proc. Int. Conf. Learn. Representations*, 2017.
[45] Z. Yang, X. He, J. Gao, L. Deng, and A. J. Smola, "Stacked attention networks for image question answering," in *Proc. IEEE Conf. Comput. Vision Pattern Recognit.*, 2016, pp. 21–29.
[46] Y. Zhu, J. J. Lim, and L. Fei-Fei, "Knowledge acquisition for visual question answering via iterative querying," in *Proc. IEEE Conf. Comput. Vision Pattern Recognit.*, 2017, pp. 6146–6155.
[47] M. Wang, Z. Lu, H. Li, and Q. Liu, "Memory-enhanced decoder for neural machine translation," in *Proc. Conf. Empirical Methods Natural Lang. Process.*, 2016, pp. 278–286.
[48] J. Cheng, L. Dong, and M. Lapata, "Long short-term memory-networks for machine reading," in *Proc. Conf. Empirical Methods Natural Lang. Process.*, 2016, pp. 551–561.
[49] I. J. Goodfellow *et al.*, "Generative adversarial networks," 2014, arXiv:1406.2661.
[50] H. Zhu, M. Long, J. Wang, and Y. Cao, "Deep hashing network for efficient similarity retrieval," in *Proc. 30th AAAI Conf. Artif. Intell.*, 2016, pp. 2415–2421.
[51] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vision Pattern Recognit.*, 2016, pp. 770–778.
[52] R. Kiros *et al.*, "Skip-thought vectors," in *Proc. 28th Int. Conf. Neural Inf. Process. Syst.*, 2015, pp. 3294–3302.
[53] B. Zhou, À. Lapedriza, J. Xiao, A. Torralba, and A. Oliva, "Learning deep features for scene recognition using places database," in *Proc. 27th Int. Conf. Neural Inf. Process. Syst.*, 2014, pp. 487–495.
[54] K. Järvelin and J. Kekäläinen, "IR evaluation methods for retrieving highly relevant documents," in *Proc. Int. ACM SIGIR Conf. Res. Develop. Inf. Retrieval*, 2000, pp. 41–48.
[55] A. Karpathy and F. F. Li, "Deep visual-semantic alignments for generating image descriptions," in *Proc. IEEE Conf. Comput. Vision Pattern Recognit.*, 2015, pp. 3128–3137.
[56] K. Wang, R. He, W. Wang, L. Wang, and T. Tan, "Learning coupled feature spaces for cross-modal matching," in *Proc. IEEE Int. Conf. Comput. Vision*, 2013, pp. 2088–2095.
[57] Y. Jia *et al.*, "Caffe: Convolutional architecture for fast feature embedding," in *Proc. 22nd ACM Int. Conf. Multimedia*, 2014, pp. 675–678.

Ge Song received the B.Sc. degree in computer science and technology from Zhengzhou University, Zhengzhou, China, in 2014. He is currently working toward the Ph.D. degree at the Nanjing University of Aeronautics and Astronautics, Nanjing, China. His research interests are in image retrieval, machine learning, pattern recognition, and computer vision.

Dong Wang is currently working toward the Ph.D. degree in computer science and technology at the Nanjing University of Aeronautics and Astronautics, Nanjing, China. His research interests include robust learning with label noise, Bayesian learning, metric learning, and feature learning.

Xiaoyang Tan received the B.Sc. and M.Sc. degrees in computer applications from Nanjing University of Aeronautics and Astronautics (NUAA) in 1993 and 1996, respectively, and the Ph.D. degree from the Department of Computer Science and Technology of Nanjing University, China. Then he worked at NUAA in June 1996 as an assistant lecturer.